APPLICATION

FOR

UNITED STATES LETTERS PATENT

TITLE: REPRESENTING A SIMULATION MODEL USING A HARDWARE CONFIGURATION DATABASE

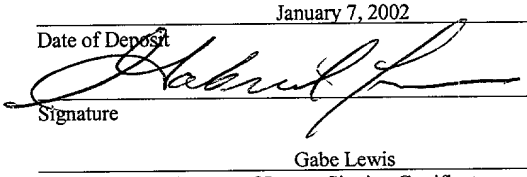
APPLICANT: TIMOTHY J. FENNELL AND WILLIAM R. WHEELER

CERTIFICATE OF MAILING BY EXPRESS MAIL

Express Mail Label No. ET 322 645 575 US

I hereby certify that this correspondence is being deposited with the United States Postal Service as Express Mail Post Office to Addressee with sufficient postage on the date indicated below and is addressed to the Commissioner for Patents, Washington, D.C. 20231.

January 7, 2002
Date of Deposit



Signature

Gabe Lewis
Typed or Printed Name of Person Signing Certificate

REPRESENTING A SIMULATION MODEL USING A HARDWARE CONFIGURATION DATABASE

CROSS-REFERENCE TO RELATED APPLICATIONS

This application is based on, and claims priority from, U.S. Provisional Application Ser. No. 60/315,852, filed August 29, 2001.

BACKGROUND

The present invention relates to representing a simulation model of an integrated circuit (chip).

A simulator employs a processor chip model to provide detailed processor chip emulation to allow a user to create system designs using the chip. The simulator model represents the processor chip and provides the user with detailed system electrical responses of the processor chip within the system. The simulated behavior allows the user to verify the predicted responses of the system implementation using the processor chip.

A graphical user interface (GUI) can allow users to write microcode, which is translated into simulator commands. The GUI can also provide visual indications of processor chip responses. Users can develop microcode for use with designs before the design is fabricated, providing a head start for microcode development efforts.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 illustrates a hardware configuration database interface between a GUI and a processor chip simulation model.

FIG. 2 illustrates an example of hierarchical levels in the hardware configuration.

FIG. 3 illustrates a GUI access to a particular hardware component in either a single- or dual-processor simulation model.

FIGS. 4A and 4B are an example of file header information used by the hardware configuration database for the single- and dual-processors, respectively, of FIG. 3.

FIG. 5. is an example of hardware configuration database code to set up access to the control store and the register for both single- and dual-processors of FIG. 3.

FIG. 6 is an example of GUI code used to access the control store and register for both single- and dual-processors of FIG. 3.

DETAILED DESCRIPTION

FIG. 1 illustrates a system that includes a processor integrated circuit (chip) simulator model 40 and a graphical user interface (GUI) 10. An intermediate hardware configuration description database 20 provides the GUI 10 with processor chip hierarchy and connectivity information. The database 20 uses a configuration language that can be interpreted by GUI 10. Using the language, the user describes

a design configuration for the processor chip simulation model 40 to GUI 10.

Once the design configuration is described, the design implementation can be queried through the GUI 10. A ``query'' is a request for information. A query instituted through the GUI may include requests for information on processor chip simulation model 40 such as contents of a memory location, status of a register, or other information desired as a processor chip design aid.

Referring to FIG. 2, the hardware configuration database 20 includes the location of hardware devices such as registers, microengines, etc. The hardware devices represent functional building blocks and sub-blocks of a processor chip design. Through use of the GUI 10 and the hardware configuration database, a user can search through the interconnections of devices to find the devices of interest.

The configuration language enables the user to provide a description of the processor simulation model 40 to GUI 10 in terms of the functional building blocks. The functional building blocks can include, but are not limited to, memories 26, control stores 22, microengines 28, 30 and registers 32.

The hardware configuration database 20 allows users to move the location of the functional blocks and subcomponents of hardware inside the simulation model 40 without requiring changes to the GUI. A query of the GUI software enables a searche for the functional blocks and subcomponents in the hardware configuration database 20 until the particular items

in the simulation model 40 are located. Once located, the GUI 10 can be used to inject and examine states of components in the simulation model 40 without requiring a hard-coded path to those simulation entities.

A user can specify connectivity of units by coupling the functional building blocks using the hardware configuration language to form higher-level groupings. The user configures one or more of these higher-level groupings into a yet higher-level component. The integrated circuit design is, thus, simulated by hierarchical levels of subcomponents. Multiple hierarchical levels may be described. The connectivity between the levels and components also can be described using the hardware configuration language. The top level of the configuration hierarchy is the GUI simulation connection.

A particular implementation uses a single hierarchical level that includes a first unit 36 and a second unit 34 as illustrated in FIG. 2. Assume, for example, that GUI 10 requires information about a control store 22. Querying the hardware configuration database 20 for control store 22 allows GUI 10 to locate control store 24 in the simulation model 40. Once the control store 24 is located, the GUI 10 can be used to operate on the control store 24 directly, without assistance from the hardware configuration database 20.

FIG. 3 illustrates that GUI 10 need not be dependent on the particular simulation processor design. In this example, GUI 10 accesses two different simulation models, 40a and 40b, through hardware design databases 20a and 20b, respectively.

Simulation model 40b is a single-processor model, and the simulation model 40a is a dual-processor model. Simulation model 40a has two microengines 310 and 312 in which register 302 is associated with microengine 310 and control store 308 is associated with microengine 312. Simulation model 40b has one microengine 314 with which both register 302 and control store 308 are associated.

The GUI 10 can find the location of register 302 and control store 308 whether the simulation is accomplished in the single-processor simulation model 40b or dual-processor simulation model 40a by implementing either the hardware configuration database 306 or 304, respectively.

The GUI needs little or no information on the simulation model stored internally to the GUI. This hardware independence indicates that the GUI 10 is less affected by hardware changes to the simulation model. Also, the same GUI software may be used to interface to multiple processor designs because the specific details of each processor design are in the hardware configuration description database, not in the GUI software.

Specific simulation data can be located by the GUI 10 in the simulation model 40 without requiring hardware specific information to be contained in the GUI. The GUI uses the hardware configuration database 20 to search the simulation model 40 and locate simulation components that can be specified in the hardware configuration database. Once located, these components may be operated on to, for example,

read and/or write simulation state information, or provide electrical stimulus and/or monitor responses in those simulation components. For example the GUI can locate control store 308 in either the single- or dual-processor simulation models. Once control store 308 is located, the GUI may access and read the data in the control store or write new data to the control store. The GUI does not need to hard-code information specifying that control store 308 is in microengine 310 in the dual-processor configuration and microengine 314 in the single-processor configuration.

The GUI is 10 useable in connection with different processor configurations. Later processor design efforts may use the same GUI with little or no change in the hardware configuration of the GUI. Also, the reusability of the GUI software means that the GUI requires less rework for each hardware design implementation or modification.

Referring to FIGS. 4A and 4B, the macros DEF_CLK, DEF_REG and DEF_ARRAY define a binding of the hierarchical path to a variable name for the single- and dual-processor simulation models, respectively. The variable name *ustore* defines the path to control store 308 which is in microengine 312 of the double-processor simulation model 40a and in microengine 314 of the single-processor simulation model 40b. Once the variable name is defined, a movement of the physical location of the control store, for example, only requires a change of the macro DEF_ARRAY to point to the new physical location in the simulation model. Both the single- and dual-processor

hardware configurations can use the same variable to describe the control store location, but each sets the variable to point to the location for its particular configuration.

FIG. 5 illustrates, in a particular example, C++ code for the hardware configuration database which will map the control store and register to the GUI. In this example, the code creates a top-level block, *chip*, and connects *chip* to a subblock, *useq*. The *useq* block is associated with two blocks: the control store and the register. The GUI 10 can locate the control store 308 in this hierarchy of blocks or it can ask for control store 308 using the previously defined label, *ustore*. The hardware configuration database 20 then can ask for the control store 308 using the name *ustore* and the simulator model can locate the control store 308 block which is *314.ram_ustore* for the single-processor and *312.ram_ustore* for the dual-processor case.

FIG. 6 illustrates, in a particular example, C++ code in the GUI that asks for the control store 308 and the register 302. In this example, the GUI is requesting specific names which were defined earlier. Other calls can find all registers that are contained in the hardware configuration database. The call illustrated, returns a handle to the register or control store specifically requested by the GUI.

Various features of the system can be implemented in a computer-implemented process and an apparatus for practicing the process. Some or all of the features of the system also

can be implemented in computer program code containing instructions embodied in tangible media, such as floppy diskettes, CD-ROMs, hard drives, or any other computer-readable storage medium. The computer program code can be loaded into and executed by a computer. The various features can also be embodied in computer program code, for example, whether stored in a storage medium, loaded into and/or executed by a computer, or transmitted over a transmission medium, such as over electrical wiring or cabling, through fiber optics, or by electromagnetic radiation. When implemented on a general-purpose microprocessor, the computer program code segments configure the microprocessor to create specific logic circuits.

Other implementations are within the scope of the following claims.